计 算 机 研 究 与 发 展
Journal of Computer Research and Development
DOI: 10.7544/issn1000-1239.202220472
60(12): 2697−2708, 2023

MIBS 分组密码的改进积分攻击

毛永霞　吴文玲　张　丽
（中国科学院软件研究所　北京　100190）
（中国科学院大学　北京　100049）
（yongxia2018@iscas.ac.cn）

Improved Integral Attacks on MIBS Block Cipher

Mao Yongxia, Wu Wenling, and Zhang Li
（*Institute of Software*, *Chinese Academy of Sciences*, *Beijing* 100190）
（*University of Chinese Academy of Sciences*, *Beijing* 100049）

Abstract　MIBS is a lightweight block cipher which was proposed by Izadi et al. at CANS 2009. Its overall encryption structure uses the typical Feistel network, and the round function adopts the SP network. MIBS supports both MIBS-64 and MIBS-80 versions, that is, it has 64-bit and 80-bit two key lengths with a 64-bit block size, and is suitable for strictly resource-constrained devices, such as low-cost RFID (radio frequency identification) tags. We study the integral attack on the block cipher MIBS. Firstly, we observe the key schedules of MIBS-64 and MIBS-80, and find some properties between their round keys by using the automatic search algorithm for key-bridging technique, respectively. Secondly, using the bit-based division property and the automatic modeling search method based on MILP (mixed integer linear programming), we find some 8-round and 9-round integral distinguishers of MIBS. Then, based on the 8-round integral distinguisher, we launch a 12-round key recovery attack for MIBS-64 with the data complexity 2^{60}, and the time complexity $2^{63.42}$. Finally, based on the 9-round integral distinguisher, we launch a 14-round key recovery attack for MIBS-80 with the data complexity 2^{63}, and the time complexity 2^{66}. These two key recoveries are the current best integral attacks on the block cipher MIBS-64 and MIBS-80.

Key words　integral attack; MIBS; key-bridging technique; partial sum technique; key recovery

摘　要　MIBS 算法是由 Izadi 等人在 CANS 2009 上提出的一个轻量级分组密码算法，整体采用 Feistel 结构，轮函数使用 SP 结构，分组长度为 64 b，包含 MIBS-64 和 MIBS-80 这 2 个版本，适用于资源受限的环境，例如 RFID（radio frequency identification）标签. 研究 MIBS 算法针对积分攻击的安全性. 首先，针对该算法的密钥编排算法，利用密钥搭桥技术，分别得到了 MIBS-64 和 MIBS-80 的轮密钥的相关性质. 其次，利用基于 MILP（mixed integer linear programming）的比特可分性的自动化建模搜索方法，构造了 MIBS 的 8 轮和 9 轮积分区分器. 然后，基于 8 轮积分区分器，给出了 12 轮 MIBS-64 的密钥恢复攻击，数据复杂度为 2^{60}，时间复杂度为 $2^{63.42}$；最后，基于 9 轮积分区分器，给出了 14 轮 MIBS-64 的密钥恢复攻击，数据复杂度为 2^{63}，时间复杂度为 2^{66}. 这是目前对 MIBS-64 和 MIBS-80 轮数最长的积分攻击.

关键词　积分攻击；MIBS；密钥搭桥技术；部分和技术；密钥恢复

中图法分类号　TP309

收稿日期：2022-06-10；**修回日期**：2022-12-09

基金项目：国家自然科学基金项目（62072445）

This work was supported by the National Natural Science Foundation of China (62072445).

近 20 年来, 伴随着物联网技术的飞速发展, 资源受限设备如低成本的 RFID(radio frequency identification)标签、无线传感器、嵌入式系统等的应用越来越广泛. 为资源受限设备研制低成本、低能耗的轻量级密码算法是一项具有挑战性的工作, 吸引了许多密码学者的关注. 例如, Leander 等人[1]设计了 DES(data encryption standard)加密算法的一个轻量级变体 DESL; Poschmann 等人[2]设计了一个比 DESL 更强的 DES 的变体 DESXL; Bogdanov 等人[3]提出了著名的轻量级密码算法 PRESENT; Banik 等人[4]设计了比 PRESENT 更安全、高效的轻量级密码算法 GIFT.

MIBS 是一个适用于资源受限环境的轻量级分组密码算法, 由 Izadi 等人[5]在 CANS 2009 会议上提出. 针对 MIBS 分组密码算法的现有分析结果包括差分分析[6]、线性分析[6]、积分分析、不可能差分分析[7]等. 目前对于 MIBS-64 最好的分析结果是 14 轮的差分分析, 成功概率为 50.15%; 对 MIBS-80 最好的分析结果是 18 轮的线性分析, 成功概率为 72.14%.

积分分析是由 Knudsen 等人[8]在 FSE 2002 上提出来的, 由于它的思想原型首先被应用于分组密码 Square[9], 因此也被称为 Square 攻击. 积分分析是当前评估分组密码算法安全性的基础分析方法之一, 已经在许多分组密码算法上得到了较好的攻击结果, 例如 AES[10], PRESENT[11]等. 可分性是积分分析的推广, 在 EUROCRYPT 2015 上被 Todo[12]提出. Todo 结合密码算法非线性部件的代数次数, 优化了积分性质, 使得积分特征能够用一种更精确的方式推导. 一个显著的应用是第一次在理论上对全轮的 MISTY1 进行了积分攻击[13]. 在 ASIACRYPT 2016 上, Xiang 等人[14]将基于混合整数线性规划(mixed integer linear programming, MILP)建模的方法引入基于比特的可分性, 进一步提高了积分区分器可自动化搜索的密码算法的规模. 2017 年, Todo 等人[15]对上述 MILP 模型进行了补充, 并将其应用于多个流密码算法的立方攻击, 得到了多个流密码当时最好的密钥恢复攻击. 近几年, 对分组密码算法的积分区分器搜索的改进主要围绕改进非线性层[16-19]和线性层[19-24]的模型进行.

目前, 对 MIBS 算法的积分分析已经存在一些结论. 2013 年, 于晓丽等人[25]基于一个 5 轮的积分区分器, 利用高阶积分技术将该区分器向前扩展 3 轮, 分别对 MIBS-64 和 MIBS-80 进行了 8, 9, 10 轮的积分攻击. 2014 年, 潘志舒等人[26]构造了 MIBS 的 5 轮积分区分器, 利用 Feistel 结构的等价结构以及 MIBS 密钥编排算法中主密钥和轮密钥的关系, 给出 10 轮 MIBS 算法的积分攻击. 2016 年, 伊文坛等人[27]利用零相关线性逼近和积分区分器之间的联系, 推导出 MIBS 的 8 轮积分区分器, 进而对 11 轮的 MIBS-80 进行了攻击. 2021 年, 李艳俊等人[28]基于 5 轮的积分区分器, 向前、向后各扩展 3 轮, 给出了一个 11 轮 MIBS-64 的积分攻击.

部分和(partial sum technique)技术是 Ferguson 等人[10]在分析 Rijndael 时提出的一种降低攻击复杂度的方法, 是改进积分攻击的有效方法. 该方法主要利用积分攻击需要对中间状态进行求和的特点, 通过压缩密钥恢复过程中的数据量来有效减少计算时间, 例如, 将对 6 轮 Rijndael 的密钥恢复时间复杂度从2^{72}降低为2^{49}[10].

本文研究 MIBS-64 和 MIBS-80 的积分分析, 主要贡献有 4 个方面:

1) 针对密钥编排算法, 利用密钥搭桥技术, 得到了轮密钥之间的一些相关性质;

2) 构造了 MIBS 的 8 轮和 9 轮积分区分器;

3) 对于 MIBS-64, 基于 8 轮的积分区分器, 在区分器末尾增加 4 轮, 给出了 12 轮的密钥恢复攻击;

4) 对于 MIBS-80, 基于 9 轮积分区分器, 在末尾增加 5 轮, 给出了 14 轮的密钥恢复攻击.

3)和 4)这 2 个攻击是目前对 MIBS-64 和 MIBS-80 已知的最好的积分攻击, 与已有积分攻击结果的对比如表 1 所示.

Table 1 The Integral Attack Results on MIBS

表 1 对 MIBS 的积分攻击结果

算法	轮数	数据复杂度	时间复杂度	参考来源
MIBS-64	10	$2^{61.6}$	2^{40}	文献 [25]
	10	2^{28}	$2^{52.7}$	文献 [26]
	11	2^{58}	$2^{59.75}$	文献 [28]
	12	2^{60}	$2^{63.42}$	本文第 3 节
MIBS-80	10	$2^{61.6}$	2^{40}	文献 [25]
	10	$2^{28.2}$	$2^{53.2}$	文献 [26]
	11	2^{60}	$2^{59.8}$	文献 [27]
	14	2^{63}	2^{66}	本文第 4 节

1 预备知识

1.1 符号说明

$C_{[i:j]}$: 密文的第 j 比特到第 i 比特;

$X_{t,[i:j]}$: 第 t 轮中间状态的第 j 比特到第 i 比特;

X_t^i: 第t轮中间状态的第i个半字节；

$sk_{t,[i:j]}$: 第t轮密钥的第j比特到第i比特；

sk_t^i: 第t轮密钥的第i个半字节；

$\boldsymbol{state}_{[i:j]}$: 中间状态的第$j$比特至第$i$比特；

S/S_t: S 盒函数/第t个 S 盒函数；

a/c: 一个活跃比特/常数比特；

$\mathcal{A}/\mathcal{C}$: 一个活跃半字节/常数半字节；

$\mathcal{B}/\mathcal{U}$: 一个平衡半字节/未知半字节；

>>>: 右循环移位操作；

||: 字符串的连接操作；

∘ 函数的复合符号.

1.2 MIBS 加密算法简介

分组密码 MIBS 整体采用 Feistel 结构，轮函数使用 SP 结构，其分组长度 64 b，MIBS 支持 64 b 和 80 b 这 2 种密钥长度，分别记为 MIBS-64 和 MIBS-80，迭代轮数为 32 轮. MIBS 中所有的运算都是基于半字节的.

设 MIBS 的轮函数为F，第i轮的轮密钥为K_i，输入为(X_i, X_{i-1})，那么输出为(X_{i+1}, X_i)，其中$X_{i+1} = F(X_i, K_i) \oplus X_{i-1}$. 轮函数$F$由异或子密钥层$AK$、S 盒层（4×4 的 S 盒）和线性变换$P$组成，记为$F = P \circ S \circ AK$，具体如图 1 所示.

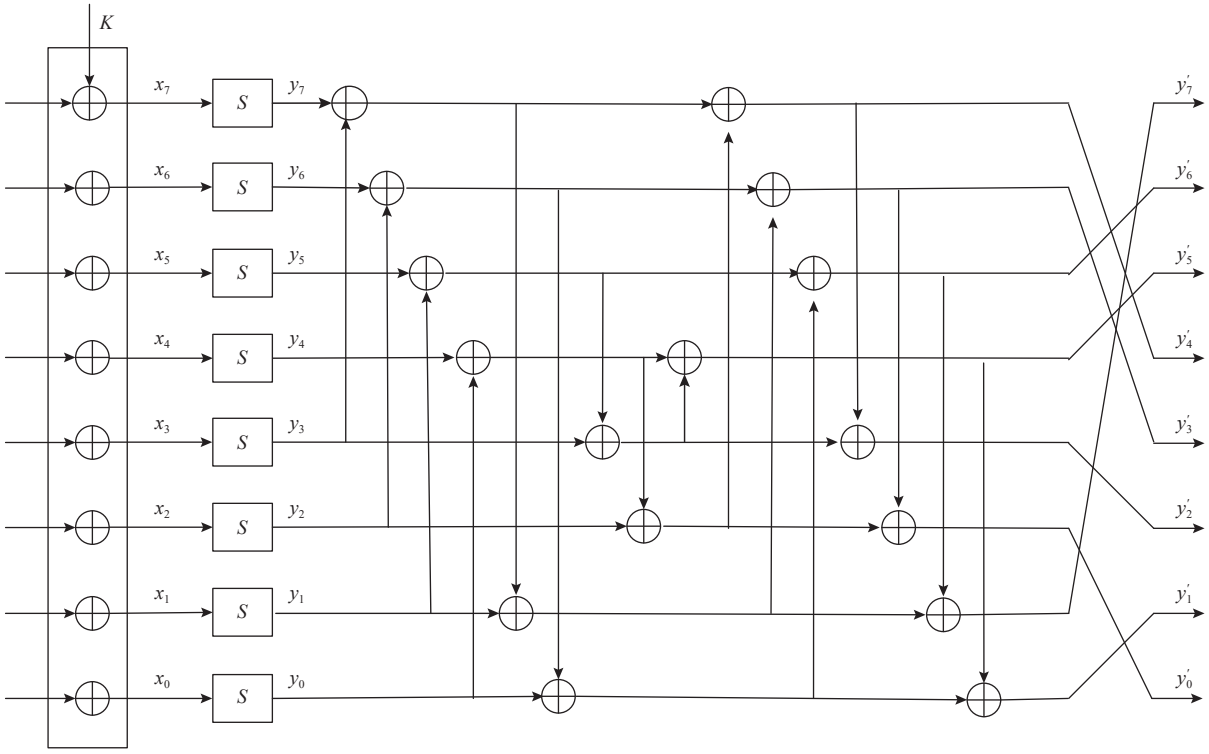


Fig. 1 Round function of MIBS

图 1 MIBS 的轮函数

设线性变换$P: (\mathbb{F}_2^4)^8 \to (\mathbb{F}_2^4)^8$的输入为$y = (y_7, y_6, y_5, y_4, y_3, y_2, y_1, y_0)$，输出为$y' = (y'_7, y'_6, y'_5, y'_4, y'_3, y'_2, y'_1, y'_0)$，那么

$$y'_7 = y_0 \oplus y_2 \oplus y_3 \oplus y_5 \oplus y_6 \oplus y_7 ,$$

$$y'_6 = y_0 \oplus y_1 \oplus y_2 \oplus y_5 \oplus y_6 ,$$

$$y'_5 = y_0 \oplus y_1 \oplus y_3 \oplus y_4 \oplus y_5 ,$$

$$y'_4 = y_0 \oplus y_2 \oplus y_3 \oplus y_4 \oplus y_7,$$

$$y'_3 = y_1 \oplus y_2 \oplus y_3 \oplus y_6 \oplus y_7,$$

$$y'_2 = y_0 \oplus y_1 \oplus y_2 \oplus y_4 \oplus y_5 \oplus y_7,$$

$$y'_1 = y_1 \oplus y_2 \oplus y_3 \oplus y_4 \oplus y_5 \oplus y_6,$$

$$y'_0 = y_0 \oplus y_1 \oplus y_3 \oplus y_4 \oplus y_6 \oplus y_7 .$$

1.3 MIBS 密钥编排算法简介

MIBS 的密钥编排采用了与 PRESENT 的密钥编排相同的设计原则.

1）MIBS-64 的密钥编排算法

设$\boldsymbol{MK}_{64}$是长度为 64 b 的主密钥，记为$\boldsymbol{MK}_{64} = (k_{63}, k_{62}, \cdots, k_0)$. 设密钥编排算法第$i$轮的中间状态为$\boldsymbol{state}^i$. 由主密钥生成长度为 32 b 的轮密钥$K_i$（$1 \leqslant i \leqslant 32$）的过程为：

$$\boldsymbol{state}^0 = \boldsymbol{MK}_{64};$$

$$\boldsymbol{state}^i \leftarrow \boldsymbol{state}^i >>> 15;$$

$$\boldsymbol{state}^i \leftarrow S(\boldsymbol{state}^i_{[63:60]}) || \boldsymbol{state}^i_{[59:0]};$$

$\boldsymbol{state}^i \leftarrow \boldsymbol{state}^i_{[63:16]} \| \boldsymbol{state}^i_{[15:11]} \oplus RC \| \boldsymbol{state}^i_{[10:0]}$;

$K_i = \boldsymbol{state}^i_{[63:32]}$.

其中RC表示轮计数, 第i轮状态$\boldsymbol{state}^i$的左 32 b 作为第i轮的轮密钥.

2)MIBS-80 的密钥编排算法

设$\boldsymbol{MK}_{80}$是长度为 80 b 的主密钥, 记为$\boldsymbol{MK}_{80} = (k_{80}, k_{79}, \cdots, k_0)$. 设密钥编排算法第$i$轮的中间状态为$\boldsymbol{state}^i$. 由主密钥生成长度为 32 b 的轮密钥$K_i$($1 \leqslant i \leqslant 32$)的过程为:

$\boldsymbol{state}^0 \leftarrow \boldsymbol{MK}_{80}$;

$\boldsymbol{state}^i \leftarrow \boldsymbol{state}^i >>> 19$;

$\boldsymbol{state}^i \leftarrow S(\boldsymbol{state}^i_{[79:76]}) \| S(\boldsymbol{state}^i_{[75:72]}) \| \boldsymbol{state}^i_{[71:0]}$;

$\boldsymbol{state}^i \leftarrow \boldsymbol{state}^i_{[79:19]} \| \boldsymbol{state}^i_{[18:14]} \oplus RC \| \boldsymbol{state}^i_{[13:0]}$;

$K_i \leftarrow \boldsymbol{state}^i_{[79:48]}$.

其中第i轮状态$\boldsymbol{state}^i$的左 32 b 作为第i轮的轮密钥.

1.4 密钥搭桥技术

密钥搭桥技术最早是在文献 [29] 中针对 AES-192 提出的. 该技术的主要作用是根据密钥编排算法, 推导出某些轮密钥之间存在的依赖关系, 即: 从某些轮密钥字节中计算出其他的轮密钥字节, 即使这些字节距离很多的扩散步骤. 在 FSE 2016 上, Lin 等人[30]提出了一个高效的比特迭代型密钥搭桥自动搜索算法, 这个算法可以改进几乎所有对分组密码攻击的复杂度.

比特迭代型密钥搭桥自动搜索算法的输入为一个表示密钥编排算法的方程系统和一个想要找到其中变量关系的密钥变量集合$\mathcal{K}_0$, 输出是密钥变量集合$\mathcal{K}_0$中存在的密钥桥, 具体过程可以划分为 2 个阶段:

1)知识传播阶段, 主要利用 Gauss-Jordan 消元法处理方程系统的系数矩阵;

2)关系导出阶段, 根据 1)处理之后矩阵的秩, 导出密钥间的线性关系, 即密钥桥.

2 MIBS 的密钥编排性质和积分区分器

2.1 MIBS 密钥编排的相关性质

针对 MIBS-64 和 MIBS-80 的密钥编排算法, 利用文献 [30] 的密钥搭桥技术搜索轮密钥之间可能存在的相关关系, 即密钥桥. 算法 1 描述了这一过程. 设$\mathcal{K}$表示 MIBS 密钥编排算法的所有中间密钥变量的集合, $\mathcal{K}_0$表示 MIBS-64 第 9~12 轮的部分轮密钥集合和 MIBS-80 第 10~14 轮的部分轮密钥集合, $\mathcal{K}_1$表示$\mathcal{K}_0$可以扩张到的集合, $\mathcal{S}$表示 S 盒的输入输出比特变量集合, $\boldsymbol{M}_{\text{key}}$表示由密钥编排算法生成的密钥方程系统$E$的系数矩阵. 矩阵中变量的顺序为$(\mathcal{K}-\mathcal{S}-\mathcal{K}_1, \mathcal{S}, \mathcal{K}_1, c)$, 其中$\mathcal{K}-\mathcal{S}-\mathcal{K}_1$表示$\mathcal{S}$和$\mathcal{K}_1$在$\mathcal{K}$中的补集, c表示常数的列. $G(\boldsymbol{M}_{\text{key}})$表示利用 Gauss-Jordan 消元法将$\boldsymbol{M}_{\text{key}}$变为对角矩阵; $G_n(\boldsymbol{M}_{\text{key}})$表示利用 Gauss-Jordan 消元法将$\boldsymbol{M}_{\text{key}}$的前$n$列变为对角矩阵. 算法 1 的目标是找到$\mathcal{K}_0$间潜在的关系式.

算法 1. MIBS 轮密钥关系搜索算法.

输入: 轮数r、种子密钥K、待测试密钥集$\mathcal{K}_0$;

输出: 密钥桥集合R.

① *Generate_key_equation*(r,K,sk): /*生成种子密钥K与轮密钥sk之间的具体关系式, 忽略常数加操作*/

② for i in range (0, r) do

③ $E(K, sk) \leftarrow$*key_schedule*(i,sk_i); /*sk_i是第i轮密钥, $sk_0 = K$, E是关于K和sk的密钥方程系统*/

④ end for

⑤ return $E(K, sk)$.

⑥ function *Propagation*($\mathcal{K}$,$\mathcal{K}_0$,$\mathcal{S}$,$\boldsymbol{M}_{\text{key}}$): /*知识传播阶段, 扩展$\mathcal{K}_0$到所有可表出的密钥*/

⑦ $\boldsymbol{M} \leftarrow G_{\mathcal{K}-\mathcal{K}_1-\mathcal{S}}(\boldsymbol{M})$; /*$\boldsymbol{M}$表示$\mathcal{K}-\mathcal{K}_1-\mathcal{S}$对应的系数矩阵*/

⑧ for *row* in $\boldsymbol{M}_{\text{key}}$ do

⑨ if 在前$|\mathcal{K}-\mathcal{K}_1-\mathcal{S}|$行有 1 个*row*属于*case* then

⑩ 移动x,$S(x)$,或者x与$S(x)$所对应的列;

⑪ $(\mathcal{K}_1, \mathcal{S}) \leftarrow (\mathcal{K}_1, \mathcal{S}) \cup \{x\}(\{S(x)\}, \{x, S(x)\})$; /*添加新变量*/

⑫ end if

⑬ end for

⑭ return $\mathcal{K}_1$, M_{key}.

⑮ function *Derivation*($\mathcal{K}_0$, $\mathcal{K}_1$, $\boldsymbol{A}$): /*关系导出阶段, 对系数矩阵$\boldsymbol{M}_{\text{key}}$中变量$\mathcal{K}_1$对应的分块矩阵$\boldsymbol{A}$进行处理*/

⑯ $\boldsymbol{A} \leftarrow G_{\mathcal{K}_1-\mathcal{K}_0-\mathcal{S}}(\boldsymbol{A})$;

⑰ for *row* in $\boldsymbol{B}_1$ do /*$\boldsymbol{B}_1$为矩阵$\boldsymbol{A}$中变量$\mathcal{S}$对应的分块矩阵*/

⑱ if $Rank(\boldsymbol{B}_1) \geqslant 4 - N$ then/*N为$\mathcal{S}$中元素在$\mathcal{K}_0$中出现的个数*/

⑲ 向$\boldsymbol{A}$中增加与φ和$x+\varphi'$相应的新行和新列;

⑳ if $Rank(\boldsymbol{B}_1) > 4 - N$ then

㉑ 向$\boldsymbol{A}$中增加与ψ和ψ'相应的新行和新列;

㉒ end if

㉓ end if

㉔ end for

㉕ $\boldsymbol{A} \leftarrow G_{\mathcal{K}_1-\mathcal{K}_0-\mathcal{S}}(\boldsymbol{A})$;

㉖ for *row* in $\boldsymbol{B}_2$ do /*$\boldsymbol{B}_2$是矩阵$\boldsymbol{A}$中变量$\mathcal{K}_0$对应的分块矩阵*/

㉗ $R \leftarrow$导出所有密钥桥;

㉘ end for

㉙ return R.

行⑨$case = \{case_1, case_2\}$, 其中:

$case_1 = (0, \cdots, 0, e_x, 0, \cdots, 0, e_{S(x)}, 0, \cdots, 0, e_{|\mathcal{K}-\mathcal{K}_1-\mathcal{S}|}, \cdots, e_n)$;

$case_2 = \{(0, \cdots, 0, e_t, e_x, 0, \cdots, 0, e_{|\mathcal{K}-\mathcal{K}_1-\mathcal{S}|}, \cdots, e_n), (0, \cdots, 0, e_s, e_{S(x)}, 0, \cdots, 0, e'_{|\mathcal{K}-\mathcal{K}_1-\mathcal{S}|}, \cdots, e'_n), t \neq s, e_t = e_s = 0, e_j = c \cdot e'_j, j = |\mathcal{K}-\mathcal{K}_1-\mathcal{S}|, \cdots, n-1\}$.

行⑲的φ为$\mathcal{S}$中剩余的$8-Rank(\boldsymbol{B}_1)-N$个变量, φ'是关于φ的线性组合.

行㉑的ψ为$\mathcal{S}$中剩余的$Rank(\boldsymbol{B}_1)-4+N$个变量, ψ'是关于ψ的线性组合, 它们用于表示可以由S扩展得到的密钥比特.

利用算法 1, 搜索到了 MIBS-64 和 MIBS-80 的密钥编排的相关性质——性质 1 和性质 2. 此外, 我们也通过密钥编排算法推导验证了性质 1 和性质 2.

性质 1. 根据 MIBS-64 的密钥编排算法, 第 11 轮的密钥$sk_{11,[31:17]}$可由第 12 轮密钥$sk_{12,[16:02]}$得到; 第 10 轮密钥$sk_{10,[31:30]}$可由第 12 轮密钥$sk_{12,[01:00]}$得到; 第 10 轮密钥$sk_{10,[31:15]}$可由第 11 轮密钥$sk_{11,[16:00]}$得到; 第 9 轮的轮密钥$sk_{9,[12:00]}$可由第 12 轮轮密钥$sk_{12,[31:19]}$得到, 第 9 轮的轮密钥$sk_{9,[31:15]}$可由第 10 轮的轮密钥$sk_{10,[16:00]}$得到.

性质 2. 根据 MIBS-80 的密钥编排算法, 第 13 轮的轮密钥$sk_{13,[31:19]}$可由第 14 轮的轮密钥$sk_{14,[12:00]}$得到; 第 12 轮密钥$sk_{12,[31:19]}$可由第 13 轮的轮密钥$sk_{13,[12:00]}$得到; 第 11 轮的密钥$sk_{11,[31:19]}$可由第 12 轮密钥$sk_{12,[12:00]}$得到; 第 11 轮的密钥$sk_{11,[08:00]}$可由第 14 轮密钥$sk_{14,[31:23]}$得到; 第 10 轮密钥$sk_{10,[31:19]}$可由第 11 轮密钥$sk_{11,[12:00]}$得到; 第 10 轮密钥$sk_{10,[27:00]}$可由第 14 轮密钥$sk_{14,[31:04]}$得到.

文献 [26] 提出了一个关于 MIBS-64 的轮密钥和主密钥之间关系的性质. 与密钥搭桥技术考虑所有轮密钥之间的具体关系式不同, 文献 [26] 通过密钥编排算法的循环移位操作来判断轮密钥可能涉及的所有主密钥位置. 例如, 考虑性质 1 的密钥桥$sk_{12,[16:02]} \rightarrow sk_{11,[31:17]}$. 根据文献 [26], 猜测$sk_{12,[16:02]}$需要猜测主密钥$K_{[39:20]}$, 猜测$sk_{11,[31:17]}$需要猜测主密钥$K_{[36:21]}$, 因此轮密钥$sk_{12,[16:02]}$与$sk_{11,[31:17]}$之间似乎并不能互相独立推导得出. 事实上, 根据密钥编排算法写出它们之间的表达式, 发现二者是可以互相推导的, 而这恰好是密钥搭桥技术的思想. 密钥搭桥技术通过处理轮密钥之间的具体关系式, 精准地判断出所有潜在的密钥桥.

文献 [26] 也提出了 MIBS-80 的轮密钥和主密钥之间关系的性质. 类似地, 考虑密钥桥$sk_{14,[12:00]} \rightarrow sk_{13,[31:19]}$. 那么, 根据文献 [26], 猜测$sk_{14,[12:00]}$需要猜测主密钥$K_{[79:74,09:00]}$, 猜测$sk_{13,[31:19]}$需要猜测主密钥$K_{[79:71,06:00]}$. 因此$sk_{14,[12:00]}$和$sk_{13,[31:19]}$似乎不能完全互相推导得出. 事实上, 根据轮密钥之间的关系式, $sk_{14,[12:00]}$和$sk_{13,[31:19]}$也是可以互相推导出的.

2.2 MIBS 的积分区分器

目前, MIBS 最长的积分区分器为 5 轮, 都是基于加密算法的结构特点推导求得的[25,28]. 结合文献 [14,31] 的基于比特自动化建模方法, 分别对线性变换和 S 盒生成约束条件, 建立 MIBS 的 MILP 模型, 搜索更长的积分区分器. 具体过程为: 首先, 对 MIBS 加密算法的每一个基本操作建模; 然后, 生成 MIBS 的r轮可分性传播的线性不等式系统; 接着, 给定输入和输出可分性; 最后, 利用求解器求解r轮可分性传播系统.

MIBS 的加密算法仅包含 S 盒、线性变换P、异或操作、置换操作. 对于 S 盒, 利用文献 [31] 的方法, 使用 Quine-Mclusky 方法建立约束条件. 首先, 求出 S 盒的所有可分迹, 然后利用 Logic Friday 软件生成约束这些可分迹的合取范式, 最后将合取范式转换为线性不等式(具体约束如附录 A 所示).

对于置换操作, 仅需要改变变量的位置即可. 对于异或操作$b = a_0 \oplus a_1$, 使用的建模规则为:

$$\begin{cases} a_0 + a_1 - b = 0, \\ a_0, a_1, b \in \{0, 1\}. \end{cases} \tag{1}$$

另外, 还需使用复制操作$b \rightarrow (a_0, a_1)$, 使用的建模规则为:

$$\begin{cases} b - a_0 - a_1 = 0, \\ a_0, a_1, b \in \{0, 1\}. \end{cases} \tag{2}$$

线性层中异或操作和复制操作的建模比较简单, 只需根据建模规则, 直接代入状态变量即可. 对于线性变换P, 对应的基于半字节的矩阵在附录 B 中给出. 根据此矩阵和异或操作、复制操作的建模规则, 可以直接写出其对应的基于比特的约束不等式(具体参考附录 B).

按照式(1)(2)的建模方式迭代r次, MIBS 的r轮可分性传播的线性不等式系统即可生成. 然后, 挑选仅有几比特为常数. 其余比特全活跃的输入状态, 确定输入可分性, 并且将输出对应的变量写成目标函

数, 添加到不等式系统中, 完成对 MILP 模型的建模过程. 最后, 通过求解不等式系统判断积分区分器的存在性.

根据上面的自动化建模搜索方法, 可以构造 MIBS 的 8 轮积分区分器: 从明文集的最高 32 比特选择 4 比特为常数, 其余比特都活跃, 那么经过 8 轮 MIBS 加密之后, 输出的最低 32 比特都是平衡的. 例如, 当明文的最高 4 比特为常数时, 对应的输入和输出可分性表示

$$\begin{pmatrix}\mathcal{C},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A}\\ \mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A}\end{pmatrix}\xrightarrow{8\text{轮}}\begin{pmatrix}\mathcal{U},\mathcal{U},\mathcal{U},\mathcal{U},\mathcal{U},\mathcal{U},\mathcal{U},\mathcal{U}\\ \mathcal{B},\mathcal{B},\mathcal{B},\mathcal{B},\mathcal{B},\mathcal{B},\mathcal{B},\mathcal{B}\end{pmatrix}.$$

当选择的明文集只有 1 个比特为常数时, 可以构造 9 轮积分区分器, 具体如定理 1 所示.

定理 1. 选择明文集 X, 满足最高 32 比特中的任一比特为常数, 其余比特都为活跃比特, 即对于一个固定的 $i\in\{0,1,\cdots,31\}$, $x_i=c$; 对于任意 $j\neq i$, $x_j=a$. 那么, 该明文集经过 9 轮 MIBS 加密之后, 输出的最低 32 比特都是平衡的, 即输出值的求和有形式为:

$$\begin{pmatrix}\mathcal{U},\mathcal{U},\mathcal{U},\mathcal{U},\mathcal{U},\mathcal{U},\mathcal{U},\mathcal{U}\\ \mathcal{B},\mathcal{B},\mathcal{B},\mathcal{B},\mathcal{B},\mathcal{B},\mathcal{B},\mathcal{B}\end{pmatrix}.$$

设每一轮的输入可分性 $\mathcal{D}=\mathcal{D}_{[63:32]}\|\mathcal{D}_{[31:00]}$, 表 2 展示了当最高比特为常数时, 9 轮积分区分器每一轮中间状态对应的可分性.

Table 2 Division Property of Intermediate States for the 9-Round Integral Distinguisher of MIBS

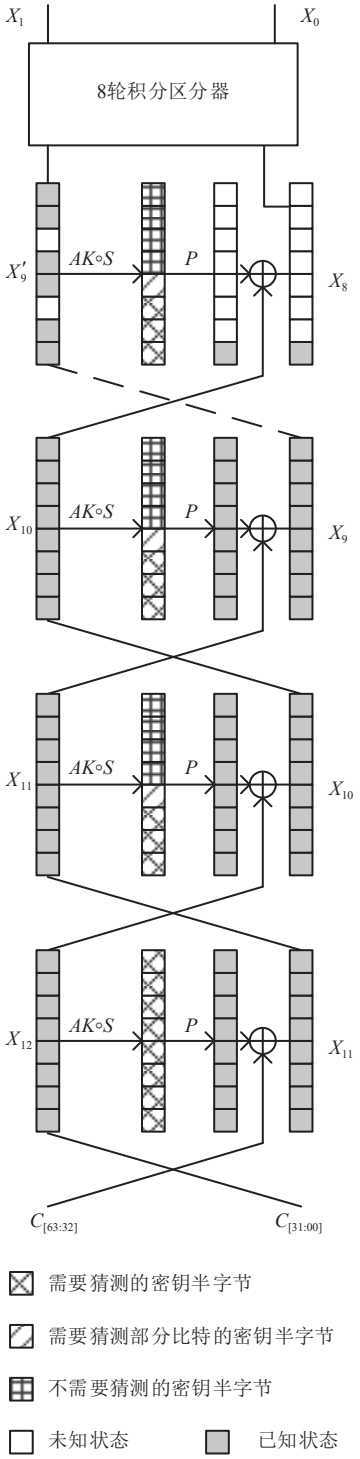
表 2 MIBS 9 轮积分区分器中间状态的可分性

轮数	左 32 b 可分性$\mathcal{D}_{[63:32]}$	右 32 b 可分性$\mathcal{D}_{[31:00]}$
0	$z\mathcal{I}_{31}$	$\mathcal{I}_{32}$
1	$\mathcal{I}_{32}$	$z\mathcal{I}_{31}$
2	$\mathcal{I}_{32}$	$i\mathcal{Z}_3\mathcal{I}_{28}$
3	$\mathcal{I}_{32}$	$i\mathcal{Z}_5iz\mathcal{I}_4zizz\mathcal{I}_{16}$
4	$\mathcal{I}_5z\mathcal{I}_{26}$	$zizzi\mathcal{Z}_5iiziziz_3i\mathcal{Z}i\mathcal{Z}_4iizzi$
5	$\mathcal{I}_3ziziiz\mathcal{I}_5ziiz\mathcal{I}_3ziizz\mathcal{I}_3zzi$	$\mathcal{Z}_3i\mathcal{Z}_6i\mathcal{Z}_3i\mathcal{Z}_5i\mathcal{Z}_4i\mathcal{Z}_3izz$
6	$zizi\mathcal{Z}_6i\mathcal{Z}_3i\mathcal{Z}_5\mathcal{I}_3zz\mathcal{I}_5zi$	$\mathcal{Z}_{32}$
7	$\mathcal{Z}_8izii\mathcal{Z}_9\mathcal{I}_3\mathcal{Z}_8$	$\mathcal{Z}_{32}$
8	$\mathcal{Z}_3i\mathcal{Z}_{21}i\mathcal{Z}_6$	$\mathcal{Z}_{32}$
9	$\mathcal{Z}_{32}$	$\mathcal{Z}_3i\mathcal{Z}_{21}i\mathcal{Z}_6$

注: $\mathcal{I}_m$,$\mathcal{Z}_m$ 表示连续$m(m\geqslant 3)$比特的可分性, 分别对应$(1,1,\cdots,1)$, $(0,0,\cdots,0)$; i, z表示基于比特的可分性, 分别对应1和0. 例如, $\mathcal{Z}_8izii$ 表示可分性$(0,0,0,0,0,0,0,0,1,0,1,1)$.

3 12 轮 MIBS-64 的密钥恢复攻击

本节利用 2.2 节中 MIBS 的 8 轮积分区分器: 最高 4 位是常数, 其余位均活跃, 8 轮之后输出的最低 32 位是平衡的, 在区分器末尾增加 4 轮, 完成了对 12 轮 MIBS-64 的密钥恢复攻击, 如图 2 所示.



X_9与X'_9之间的虚线表示: 为了计算X_8^0, 从第 10 轮X_9中选取部分状态, 即$X_9^{0,1,3,4,6,7}$参与计算

Fig. 2 Key recovery attack of MIBS-64

图 2 MIBS-64 的密钥恢复攻击

3.1 攻击过程

整个密钥恢复过程分为 6 步, 具体攻击步骤为:

Step1. 选择明文集 $X=X_1\|X_0$, X 共 2^{60} 个明文，满足明文的第 1 个半字节为常数，其余半字节都是活跃的，查询其 12 轮 MIBS-64 加密之后的密文 $C=C_{[63:32]}\|C_{[31:00]}$.

Step2. 已知密文 C，猜测第 12 轮的密钥 $sk_{12,[31:00]}$，解密计算 $X_{11}=P\circ S(C_{[63:32]}\oplus sk_{12})\oplus C_{[31:00]}$.

Step3. 已知密文和 X_{11}，猜测第 11 轮的轮密钥 $sk_{11,[31:00]}$，计算 $X_{10}=P\circ S(X_{11}\oplus sk_{11})\oplus C_{[63:32]}$. 根据性质 1，部分密钥可以根据 12 轮的轮密钥直接得到，因此实际只需猜测 15 b，即 $sk_{11,[14:00]}$.

Step4. 已知 X_{11}和 X_{10}，猜测第 10 轮的轮密钥 $sk_{10,[31:00]}$，计算 $X_9=P\circ S(X_{10}\oplus sk_{10})\oplus X_{11}$. 根据性质 1，部分密钥可以根据第 12,11 轮的轮密钥直接得到，因此实际只需猜测 15b，即 $sk_{10,[14:00]}$.

Step5. 已知 X_{10}和 X_9，猜测第 9 轮的轮密钥 $sk_{9,[31:00]}$，计算 $X_8=P\circ S(X_9\oplus sk_9)\oplus X_{10}$. 根据性质 1，部分密钥可以根据第 12,11,10 轮的轮密钥直接得到，因此实际只需猜测 15b，即 $sk_{9,[14:00]}$.

Step6. 根据 2.2 节，X 经 8 轮 MIBS 加密之后的低 32 比特都是平衡比特. 因此，满足 $\sum_i X_{i,8}=0$ 的密钥可能为正确密钥. 筛选概率为 2^{-32}，共得到 $2^{32+15+15+15}\times 2^{-32}=2^{45}$ 个候选密钥.

3.2 利用密钥搭桥与部分和技术降低复杂度

3.1 节的攻击过程已经根据密钥桥调整了需要猜测的密钥个数，下面介绍利用部分和技术降低复杂度.

对于 Step2，利用部分和技术，逐个计算 X_{11}中每半字节的值. 由于 $S(C_{[63:32]}\oplus sk_{12})=S_7(C_{[63:60]}\oplus sk_{12,[31:28]})\|\cdots\|S_0(C_{[35:32]}\oplus sk_{12,[03:00]})$，根据线性变换 P，有式(3)~(10)成立：

$$X_{11}^0=C_{[03:00]}\oplus\sum_{i\in\{0,1,3,4,6,7\}}S_i(C_{[31:00]}^i\oplus sk_{12}^i),\tag{3}$$

$$X_{11}^1=C_{[07:04]}\oplus\sum_{i\in\{1,2,3,4,5,6\}}S_i(C_{[31:00]}^i\oplus sk_{12}^i),\tag{4}$$

$$X_{11}^2=C_{[11:08]}\oplus\sum_{i\in\{0,1,2,4,5,7\}}S_i(C_{[31:00]}^i\oplus sk_{12}^i),\tag{5}$$

$$X_{11}^3=C_{[15:12]}\oplus\sum_{i\in\{1,2,3,6,7\}}S_i(C_{[31:00]}^i\oplus sk_{12}^i),\tag{6}$$

$$X_{11}^4=C_{[19:16]}\oplus\sum_{i\in\{0,2,3,4,7\}}S_i(C_{[31:00]}^i\oplus sk_{12}^i),\tag{7}$$

$$X_{11}^5=C_{[23:20]}\oplus\sum_{i\in\{0,1,3,4,5\}}S_i(C_{[31:00]}^i\oplus sk_{12}^i),\tag{8}$$

$$X_{11}^6=C_{[27:24]}\oplus\sum_{i\in\{0,1,2,5,6\}}S_i(C_{[31:00]}^i\oplus sk_{12}^i),\tag{9}$$

$$X_{11}^7=C_{[31:28]}\oplus\sum_{i\in\{0,2,3,5,6,7\}}S_i(C_{[31:00]}^i\oplus sk_{12}^i).\tag{10}$$

1)根据式(3)计算 X_{11}^0. 首先猜测 sk_{12}^0和 sk_{12}^1，解密计算求和 $S_0(C_{[31:00]}^0\oplus sk_{12}^0)\oplus S_1(C_{[31:00]}^1\oplus sk_{12}^1)$出现的次数；接着猜测 sk_{12}^3，计算第 0, 1, 3 个 S 盒的求和值出现的次数；以此类推，依次猜测 sk_{12}^4, sk_{12}^6, sk_{12}^7，最终计算得到 X_{11}^0每个可能值出现的次数. 实际上，因为解密得到的所有值最终要求异或和，所以只需统计每个值出现的次数是奇数还是偶数即可.

2)根据式(4)计算 X_{11}^1. 由于在 1)中已经猜测了 $sk_{12}^{1,3,4,6}$，因此这里只需再猜测 sk_{12}^2和 sk_{12}^5的值. 通过猜测的密钥，统计 X_{11}^1的所有可能值出现次数的奇偶性.

3)以此类推，根据猜测的密钥和式(5)~(10)，计算 X_{11}其他半字节的值.

对于 Step3，只需要猜测 $sk_{11,[14:00]}$. 与 Step2 类似，首先猜测 $sk_{11}^{0,1}$，计算 $S_0(X_{11}^0\oplus sk_{11}^0)\oplus S_1(X_{11}^1\oplus sk_{11}^1)$；接着猜测 sk_{11}^2, $sk_{11,[14:12]}$，计算 X_{10}每个半字节的值.

Step4, Step5 的过程与 Step3 类似. 图 2 也展示了第 9 轮利用部分和与密钥搭桥技术恢复 X_8^0的过程，即 $X_8^0=X_{10}^0\oplus\sum_{i\in\{0,1,3,4,6,7\}}S_i(X_9^i\oplus sk_9^i)$ X_9与 X_9'之间的虚线表示：为了计算 X_8^0，从第 10 轮 X_9中选取部分状态，即 $X_9^{0,1,3,4,6,7}$参与计算.

3.3 复杂度分析

3.1 节的攻击过程的时间复杂度主要来自于 Step2~5.

Step2 的时间复杂度约为 $2^{60}\times 2^8+2^{60}\times 2^4\times 6$次 S 盒查表；Step3 的时间复杂度约为 $2^{60}\times 2^8+2^{60}\times 2^4\times 1.75$次 S 盒查表；Step4 和 Step5 的时间复杂度都为 $2^{60}\times 2^8+2^{60}\times 2^4\times 1.75$次 S 盒查表. 因此总的时间复杂度为 $2^{60}\times 2^8\times 4/(12\times 8)\approx 2^{63.42}$次 12 轮 MIBS-64 解密计算. 数据复杂度为 2^{60}的选择明文.

4 14 轮 MIBS-80 的密钥恢复攻击

利用 2.2 节中 MIBS 的任意一个 9 轮积分区分器，在末尾增加 5 轮，都能完成一个对 14 轮 MIBS-80 的密钥恢复攻击. 本节以 MIBS 所示的一个区分器为例，介绍详细的攻击过程如图 3 所示.

$$\begin{pmatrix} caaa,\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A}\\ \mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A}\end{pmatrix}\xrightarrow{9\text{轮}}$$

$$\begin{pmatrix}\mathcal{U},\mathcal{U},\mathcal{U},\mathcal{U},\mathcal{U},\mathcal{U},\mathcal{U},\mathcal{U}\\ \mathcal{B},\mathcal{B},\mathcal{B},\mathcal{B},\mathcal{B},\mathcal{B},\mathcal{B},\mathcal{B}\end{pmatrix}.$$

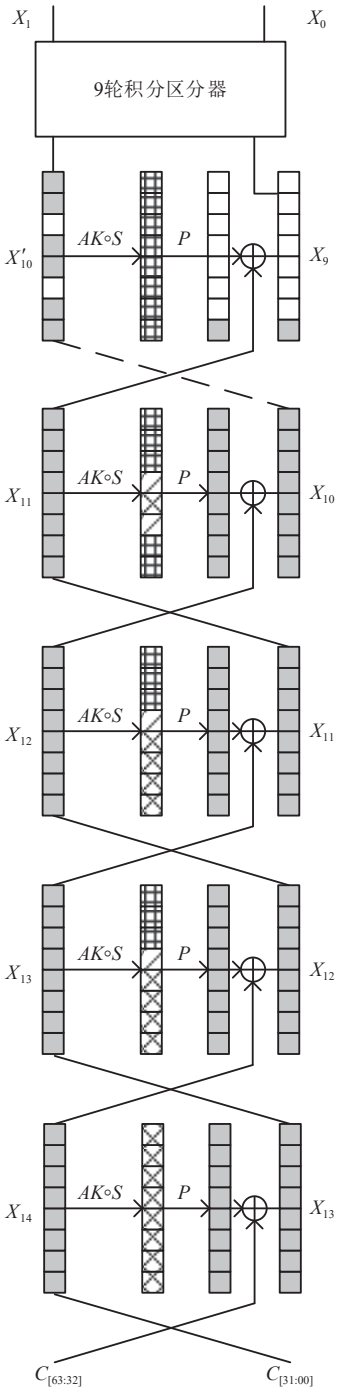


Fig. 3 Key recovery attack of MIBS-80

图 3 MIBS-80 的密钥恢复攻击

4.1 攻击过程

Step1. 选择明文集X, X中共有2^{63}个明文, 满足明文的第 1 个比特为常数, 其余比特都是活跃的, 查询其 14 轮 MIBS-80 加密之后的密文$C = C_{[63:32]}||C_{[31:00]}$.

Step2. 已知密文C, 猜测第 14 轮的密钥sk_{14}, 解密计算$X_{13} = P \circ S(C_{[63:32]} \oplus sk_{14}) \oplus C_{[31:00]}$.

Step3. 猜测第 13 轮的轮密钥sk_{13}, 解密计算$X_{12} = P \circ S(X_{13} \oplus sk_{13}) \oplus X_{14}$. 根据性质 2, 部分密钥比特可以根据$sk_{14,[12:00]}$直接得到, 因此实际只需猜测 19 b, 即$sk_{13,[18:00]}$.

Step4. 猜测第 12 轮的轮密钥sk_{12}, 解密计算$X_{11} = P \circ S(X_{12} \oplus sk_{12}) \oplus X_{13}$. 根据性质 2, 部分密钥可以根据$sk_{13,[12:00]}$直接得到, 因此实际只需猜测 19 b, 即$sk_{12,[18:00]}$.

Step5. 猜测第 11 轮的轮密钥sk_{11}, 解密计算$X_{10} = P \circ S(X_{11} \oplus sk_{11}) \oplus X_{12}$. 根据性质 2, 部分密钥可以根据 13,12 轮的轮密钥得到, 实际上只需要猜测 10 b, 即$sk_{11,[18:09]}$.

Step6. 根据性质 2, sk_{10}可由$sk_{11,[12:00]}$和$sk_{14,[31:04]}$得到, 解密计算$X_9 = P \circ S(X_{10} \oplus sk_{10}) \oplus X_{11}$.

Step7. 根据定理 1, X经 9 轮 MIBS 加密之后的低 32 比特都是平衡比特. 因此, 满足$\sum_i X_{i,9} = 0$的密钥可能为正确密钥. 筛选概率为2^{-32}, 共得到$2^{32+19+19+10} \times 2^{-32} = 2^{48}$个候选密钥.

4.2 利用密钥搭桥与部分和技术降低复杂度

4.1 节中的攻击过程已经根据密钥桥调整了需要猜测的密钥个数, 下面介绍利用部分和技术降低复杂度.

对于 Step2, 利用部分和技术, 逐个计算X_{13}每半字节的值. 根据线性变换P, 有式(11)(12)成立:

$$X_{13}^0 = C_{[03:00]} \oplus \sum_{i \in \{0,1,3,4,6,7\}} S_i(C_{[31:00]}^i \oplus sk_{14}^i), \tag{11}$$

$$X_{13}^1 = C_{[07:04]} \oplus \sum_{i \in \{1,2,3,4,5,6\}} S_i(C_{[31:00]}^i \oplus sk_{14}^i). \tag{12}$$

1)根据式(11), 计算X_{13}^0. 首先猜测sk_{14}^0和sk_{14}^1的值, 由于密文已知, 解密计算$S_0(C_{[31:00]}^0 \oplus sk_{14}^0) \oplus S_1(C_{[31:00]}^1 \oplus sk_{14}^1)$出现的次数; 接着猜测$sk_{14}^3$, 计算第 0, 1, 3 个 S 盒的求和值出现的次数; 以此类推, 依次猜测sk_{14}^4, sk_{14}^6, sk_{14}^7, 最终计算得到X_{13}^0每个可能值出现次数的奇偶性.

2)根据式(12), 计算X_{13}^1. 由于在 1)中已经猜测了$sk_{14}^{1,3,4,6}$, 因此这里只需再猜测sk_{14}^2和sk_{14}^5的值. 通过猜测的密钥, 统计X_{13}^1每个可能值出现次数的奇偶性.

3)类似于 1)和 2), 根据猜测的密钥及线性变换P计算X_{13}其他半字节的值.

Step3~6 的过程与 Step2 类似, 区别在于需要猜测的密钥长度小于 32 b.

4.3 复杂度分析

4.1 节中的攻击过程的时间复杂度主要来自于 Step2~5. Step2 的时间复杂度约为$2^{63}\times 2^{8}+2^{63}\times 2^{4}\times 6$次 S 盒查表，Step3 和 Step4 的时间复杂度都为$2^{63}\times 2^{8}+2^{63}\times 2^{4}\times 2.75$次 S 盒查表，Step5 的时间复杂度约为$2^{63}\times 2^{7}+2^{63}\times 2^{4}\times 0.75$次 S 盒查表，因此总的时间约为$2^{63}\times 2^{8}\times 3.5/14\times 8\approx 2^{66}$次 14 轮的 MIBS-80 解密计算，数据复杂度为2^{63}的选择明文.

5 总　　结

本文对分组密码算法 MIBS 进行了积分分析. 对于 MIBS-64，在 8 轮积分区分器的末尾增加 4 轮进行了 12 轮 MIBS-64 的密钥恢复攻击；对于 MIBS-80，在 9 轮积分区分器的末尾增加 5 轮进行了 14 轮 MIBS-80 的密钥恢复攻击. 在 MIBS 的密钥恢复过程中，我们利用密钥搭桥技术与部分和技术，有效地降低了时间复杂度. MIBS 的密钥编排算法比较简单，例如，MIBS-64 的 2 轮实际上只用了 47 b 互不相关的轮密钥. 因此，利用密钥搭桥技术可以极大降低猜测的密钥量. 由此可见，轮密钥之间的关系对算法的安全性具有重要的影响.

作者贡献声明：毛永霞提出实验方案并撰写论文；吴文玲提出指导意见并修改论文；张丽负责修改论文.

参　考　文　献

[1] Leander G, Paar C, Poschmann A, et al. New lightweight DES variants [C] //Proc of the 14th Int Conf on Fast Software Encryption. Berlin: Springer, 2007: 196−210

[2] Poschmann A, Leander G, Schramm K, et al. A family of light-weight block ciphers based on DES suited for RFID applications [C/OL] //Proc of Conf on RFID Security. Berlin: Springer, 2006[2022-10-31]. https://www.semanticscholar.org/paper/A-Family-of-Light-Weight-Block-Ciphers-Based-on-DES-Poschmann-Leander/4788ca1dec5495c0c17da5f2e80831acca0abca2

[3] Bogdanov A, Knudsen L R, Leander G, et al. PRESENT: An ultra-lightweight block cipher [C] //Proc of the 9th Int Conf on Cryptographic Hardware and Embedded Systems. Berlin: Springer, 2007: 450−466

[4] Banik S, Pandey S K, Peyrin T, et al. GIFT: A small present [C] //Proc of the 19th Int Conf on Cryptographic Hardware and Embedded Systems. Berlin: Springer, 2017: 321−345

[5] Izadi M, Sadeghiyan B, Sadeghian S S, et al. MIBS: A new lightweight block cipher [C] //Proc of the 8th Int Conf on Cryptology and Network Security (CANS 2009). Berlin: Springer, 2009: 334−348

[6] Bay A, Nakahara J, Vaudenay S. Cryptanalysis of reduced-round MIBS block cipher [C/OL] //Proc of the 9th Int Conf on Cryptology and Network Security 2010. Berlin: Springer, 2010[2022-10-31]. https://doi.org/10.1007/978−3-642−17619-7_1

[7] Luo Yiyuan, Lai Xuejia. Improvements for finding impossible differentials of block cipher structures[J/OL]. Security and Communication Networks, 2017 [2022-10-31]. https://ia.cr/2017/1209

[8] Knudsen L, Wagner D. Integral cryptanalysis [C] //Proc of the 9th Int Conf on Fast Software Encryption (FSE 2002). Berlin: Springer, 2002: 112−127

[9] Daemen J, Knudsen L, Rijmen V. The block cipher Square [C] //Proc of the 4th Int Conf on Fast Software Encryption (FSE 1997). Berlin: Springer, 1997: 149−165

[10] Ferguson N, Kelsey J, Lucks S, et al. Improved cryptanalysis of Rijndael [C] //Proc of the 7th Int Conf on Fast Software Encryption (FSE 2000). Berlin: Springer, 2000: 213−230

[11] Todo Y, Morii M. Compact representation for division property [C] // Proc of the 15th Int Conf on Cryptology and Network Security (CANS 2016). Berlin: Springer, 2016: 19−35

[12] Todo Y. Structural evaluation by generalized integral property [G] // LNCS 9056: Proc of EUROCRYPT 2015. Berlin: Springer, 2015: 287−314

[13] Todo Y. Integral cryptanalysis on full MISTY1 [G] //LNCS 9215: Proc of CRYPTO 2015. Berlin: Springer, 2015: 412−432

[14] Xiang Zejun, Zhang Wentao, Bao Zhenzhen, et al. Applying MILP method to searching integral distinguishers based on division property for 6 lightweight block ciphers [G] //LNCS 10031: Proc of ASIACRYPT 2016. Berlin: Springer, 2016: 648−678

[15] Todo Y, Isobe T, Hao Yonglin, et al. Cube attacks on non-blackbox polynomials based on division property [G] //LNCS 10403: Proc of CRYPTO 2017. Berlin: Springer, 2017: 250−279

[16] Sasaki Y, Todo Y. New algorithm for modeling S-box in MILP based differential and division trail search [C] //Proc of the 10th Int Conf on Innovative Security Solutions for Information Technology and Communications. Berlin: Springer, 2017: 150−165

[17] Udovenko A. Convexity of division property transitions: Theory, algorithms and compact models [G] //LNCS 13090: Proc of ASIACRYPT 2021. Berlin: Springer, 2021: 332−361

[18] Beierle C, Biryukov A, Santos LC, et al. Alzette: A 64-Bit ARX-box [G] //LNCS 12172: Proc of CRYPTO 2020. Berlin: Springer, 2020: 419−448

[19] Derbez P, Lambin B. Fast MILP models for division property[J]. IACR Transactions on Symmetric Cryptology, 2022, 2022(2): 289−321

[20] Sun Ling, Wang Wei, Wang Meiqin. MILP-aided bit-based division property for primitives with non-bit-permutation linear layers[J]. IET Information Security, 2020, 1(14): 12−20

[21] Zhang Wenying, Rijmen V. Division cryptanalysis of block ciphers with a binary diffusion layer[J]. IET Information Security, 2019, 2(13): 87−95

[22] Hu Kai, Wang Qingju, Wang Meiqin. Finding bit-based division property for ciphers with complex linear layers[J]. IACR Transactions Symmetric Cryptology, 2020, 2020(1): 396−424

[23] Hong Chunlei, Zhang Shasha, Chen Siwei, et al. More accurate division property propagations based on optimized implementations of linear layers [C] //Proc of the 17th Int Conf on Information Security and Cryptology 2021. Berlin: Springer, 2021: 212−232

[24] Elsheikh M, Youssef A M. On MILP-based automatic search for bit-based division property for ciphers with (large) linear layers [C] // Proc of the 26th Australasian Conf on Information Security and Privacy 2021. Berlin: Springer, 2021: 111−131

[25] Yu Xiaoli, Wu Wenling, Li Yanjun. Integral attack of reduced-round MIBS block cipher[J]. Journal of Computer Research and Development, 2013, 50(10): 2117−2125 (in Chinese)
(于晓丽, 吴文玲, 李艳俊. 低轮MIBS 分组密码的积分分析[J]. 计算机研究与发展, 2013, 50(10): 2117−2125)

[26] Pan Zhishu, Guo Jiansheng, Cao Jinke, et al. Integral attack on MIBS block cipher[J]. Journal on Communications, 2014, 35(7): 157−163 (in Chinese)
(潘志舒, 郭建胜, 曹进克, 等. MIBS算法的积分攻击[J]. 通信学报, 2014, 35(7): 157−163)

[27] Yi Wentan, Lu Linzhen, Chen Shaozhen. Integral and zero-correlation linear cryptanalysis of lightweight block cipher MIBS[J]. Journal of Electronics & Information Technology, 2016, 38(4): 819−826 (in Chinese)
(伊文坛, 鲁林真, 陈少真. 轻量级密码算法MIBS的零相关和积分分析[J]. 电子与信息学报, 2016, 38(4): 819−826)

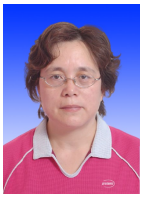
[28] Li Yanjun, Sun Qilong, Ou Haiwen, el al. Improved integral attacks on MIBS-64 block cipher[J]. Journal of Cryptologic Research, 2021, 8(4): 669−679 (in Chinese)
(李艳俊, 孙启龙, 欧海文, 等. 改进的MIBS-64算法积分分析研究[J]. 密码学报, 2021, 8(4): 669−679)

[29] Dunkelman O, Keller N, Shamir A. Improved single-key attacks on 8-round AES-192 and AES-256 [G] //LNCS 6477: Proc of ASIACRYPT 2010. Berlin: Springer, 2010: 158−176

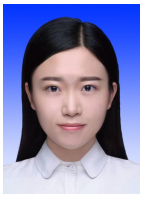
[30] Lin Li, Wu Wenling, Zheng Yafei. Automatic search for key-bridging technique: Applications to LBlock and TWINE [C] //Proc of the 23rd Int Conf on Fast Software Encryption (FSE 2016). Berlin: Springer, 2016: 247−267

[31] Abdelkhalek A, Sasaki Y, Todo Y, et al. MILP modeling for (large) S-boxes to optimize probability of differential characteristics[J]. IACR Transactions Symmetric Cryptology, 2017, 2017(4): 99−129

Mao Yongxia, born in 1988. PhD. Her main research interest includes design and analysis of block ciphers.
毛永霞, 1988 年生. 博士. 主要研究方向为分组密码的设计和分析.



Wu Wenling, born in 1966. PhD, professor, and PhD supervisor. Senior member of CCF. Her main research interest includes design and analysis of symmetric ciphers.
吴文玲, 1966 年生. 博士, 教授, 博士生导师. CCF 高级会员. 主要研究方向是对称密码的设计和分析.



Zhang Li, born in 1994. PhD. Her main research interest includes analysis of block ciphers.
张　丽, 1994 年生. 博士. 主要研究方向为分组密码的分析.

附录 A. S 盒的线性约束条件.

设$(a,b,c,d)\to(e,f,g,h)$表示 S 盒的一条可分迹，那么下面的 24 个不等式可以充分描述可分性在 MIBS 的 S 盒上的传播：

$$\begin{cases} a-e-h\geqslant -1,\\ b-f-g\geqslant -1,\\ d-e-h\geqslant -1,\\ c-f-g\geqslant -1,\\ a-f-g\geqslant -1,\\ a+b+c+d-e\geqslant 0,\\ a+b+c+d-f\geqslant 0,\\ a+b+c+d-g\geqslant 0,\\ a+b+c+d-h\geqslant 0,\\ -d+e+f+g+h\geqslant 0,\\ -a+e+f+g+h\geqslant 0,\\ -c+e+f+g+h\geqslant 0,\\ -b+e+f+g+h\geqslant 0,\\ -f+g-h\geqslant -1,\\ -e+f-g\geqslant -1,\\ -e-f+h\geqslant -1,\\ e-g-h\geqslant -1,\\ -a-d-e+h\geqslant -2,\\ -b-c-d-f+h\geqslant -3,\\ -c-e+g-h\geqslant -2,\\ -a-b-d+e-h\geqslant -3,\\ -a-b-c+f-g\geqslant -3,\\ b-e-h\geqslant -1,\\ -a-b-c-f+g\geqslant -3. \end{cases}$$

附录 B. 线性变换P的线性约束条件.

MIBS 的线性变换P对应的矩阵$\boldsymbol{M}$为

$$\boldsymbol{M}=\begin{pmatrix} 1&1&1&0&1&1&0&1\\ 0&1&1&0&0&1&1&1\\ 0&0&1&1&1&0&1&1\\ 1&0&0&1&1&1&0&1\\ 1&1&0&0&1&1&1&0\\ 1&0&1&1&0&1&1&1\\ 0&1&1&1&1&1&1&0\\ 1&1&0&1&1&0&1&1 \end{pmatrix}.$$

根据矩阵$\boldsymbol{M}$，以及复制和异或的建模规则，可以直接写出比特级线性约束条件. 以第 1 行和第 1 列为例. 设线性变换P的输入可分性$(a_{i+28}, a_{i+24}, a_{i+20}, a_{i+16}, a_{i+12}, a_{i+8}, a_{i+4}, a_i)$, $0 \leqslant i \leqslant 3$, 输出可分性$(b_{i+28}, b_{i+24}, b_{i+20}, b_{i+16}, b_{i+12}, b_{i+8}, b_{i+4}, b_i)$, $0 \leqslant i \leqslant 3$.

对于第 1 列，需要复制 5 次，设$a_{i+28} \xrightarrow{\text{复制}} (a_{i+28}^4, a_{i+28}^3, a_{i+28}^2, a_{i+28}^1, a_{i+28}^0)$. 根据复制操作的建模规则，那么第 1 列对应的约束条件可以写为

$$\begin{cases} a_{i+28} - a_{i+28}^4 - a_{i+28}^3 - a_{i+28}^2 - a_{i+28}^1 - a_{i+28}^0 = 0, \\ a_{i+28}, a_{i+28}^4, a_{i+28}^3, a_{i+28}^2, a_{i+28}^1, a_{i+28}^0 \in \{0, 1\}. \end{cases}$$

遍历i, $0 \leqslant i \leqslant 3$, 得到线性变换$P$最高 4 比特输入所对应复制操作的约束条件. 同理，遍历所有列，可得到线性变换P所有输入位置对应复制操作的约束条件.

对于第 1 行，共有 6 个输入位置可能不为 0，设$(a_{i+28}^0, a_{i+24}^0, a_{i+20}^2, a_{i+12}^0, a_{i+8}^0, a_i^0) \xrightarrow{\text{异或}} b_{i+28}$, $0 \leqslant i \leqslant 3$. 根据异或操作的建模规则，那么第 1 行对应的约束条件可以写为

$$\begin{cases} a_{i+28}^0 + a_{i+24}^0 + a_{i+20}^2 + a_{i+12}^0 + a_{i+8}^0 + a_i^0 - b_{i+28} = 0, \\ a_{i+28}^0, a_{i+24}^0, a_{i+20}^2, a_{i+12}^0, a_{i+8}^0, a_i^0, b_{i+28} \in \{0, 1\}. \end{cases}$$

遍历i, $0 \leqslant i \leqslant 3$, 得到线性变换$P$最高 4 比特输出所对应异或操作的约束条件. 同理，遍历所有行，可得到线性变换P所有输出位置对应异或操作的约束条件.